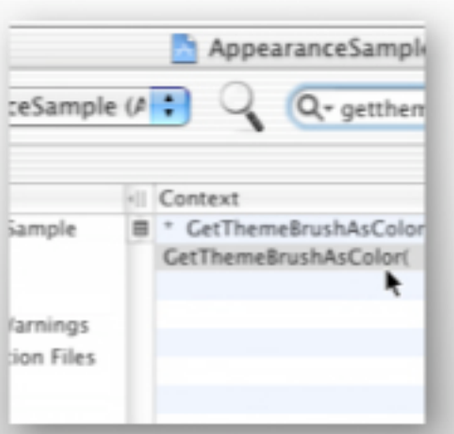
Debugging and Tuning Your Carbon Project With Xcode

Session 310



Debugging and Tuning Your Carbon Project

Tips and tricks using Xcode

Chris Espinosa
Apple Developer Tools

Introduction

- Carbon: It's not just for CodeWarrior any more
- Panther and Xcode have great support for developing and debugging Carbon apps

What You'll Learn

- Basic differences between Carbon debugging on Mac OS 9 and Mac OS X
- Techniques for using gdb and Xcode to debug Carbon apps
- New features in Xcode that make it easier to debug Carbon code
- Carbon-specific performance and tuning tips

Steps In Debugging and Tuning

- Define your quality and performance goals
- Prep your project for debugging
- Get your basic functionality working
- Run your app under the debugger to isolate bugs
- Run your app under performance analysis tools to find hot spots
- Fix bugs and tune performance until your goals are met

Basics of gdb Debugging

- Launching your app under gdb
  - gdb /path/yourapp.app/Contents/MacOS/yourapp
  - gdb your.app `ps -axwww | grep "yourapp" | head -1 | cut -c 1-5`
- The basic gdb commands
  - b sets a breakpoint
  - p displays a variable's value
  - x examines memory
  - c continues execution
  - bt prints a backtrace (stack crawl)

Carbon Debugging Aids From gdb

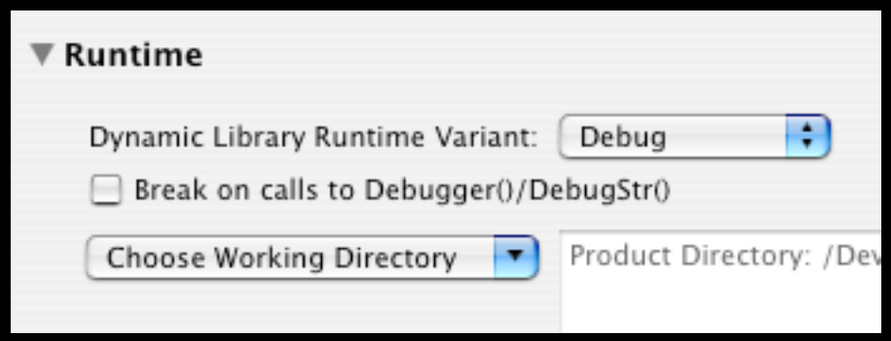
```
void HIObjectPrintDebugInfo(HIObjectRef inObject);
void DebugPrintWindowGroup(WindowGroupRef inGroup);
void DebugPrintAllWindowGroups(void) ;
void _SetEventTraceEnabled(Boolean inEnabled);
void _DebugPrintEventQueue(EventQueueRef inQueue);
void _DebugPrintEvent(EventRef inEvent);
void DebugPrintMenuList(void);
void DebugPrintMenu(MenuRef inMenu);
void DebugPrintMenuItem(MenuRef menu, MenuItemIndex item);
void DebugPrintWindow(WindowRef windowRef);
void DebugPrintWindowList(void);
void DebugFlashWindowVisRgn(WindowRef windowRef);
void DebugFlashWindowUpdateRgn(WindowRef inWindow);
```

Demo

Chris Nebel
Developer Tools Group

Using Carbon Debugging Macros

- Use Debugging.h and AssertMacros.h
  - Define DEBUG=1 and a debug level in your Debug target
  - Use verify() and kin for in-code assertions
- Using debug versions of libraries
  - Turn Zero Link off before using

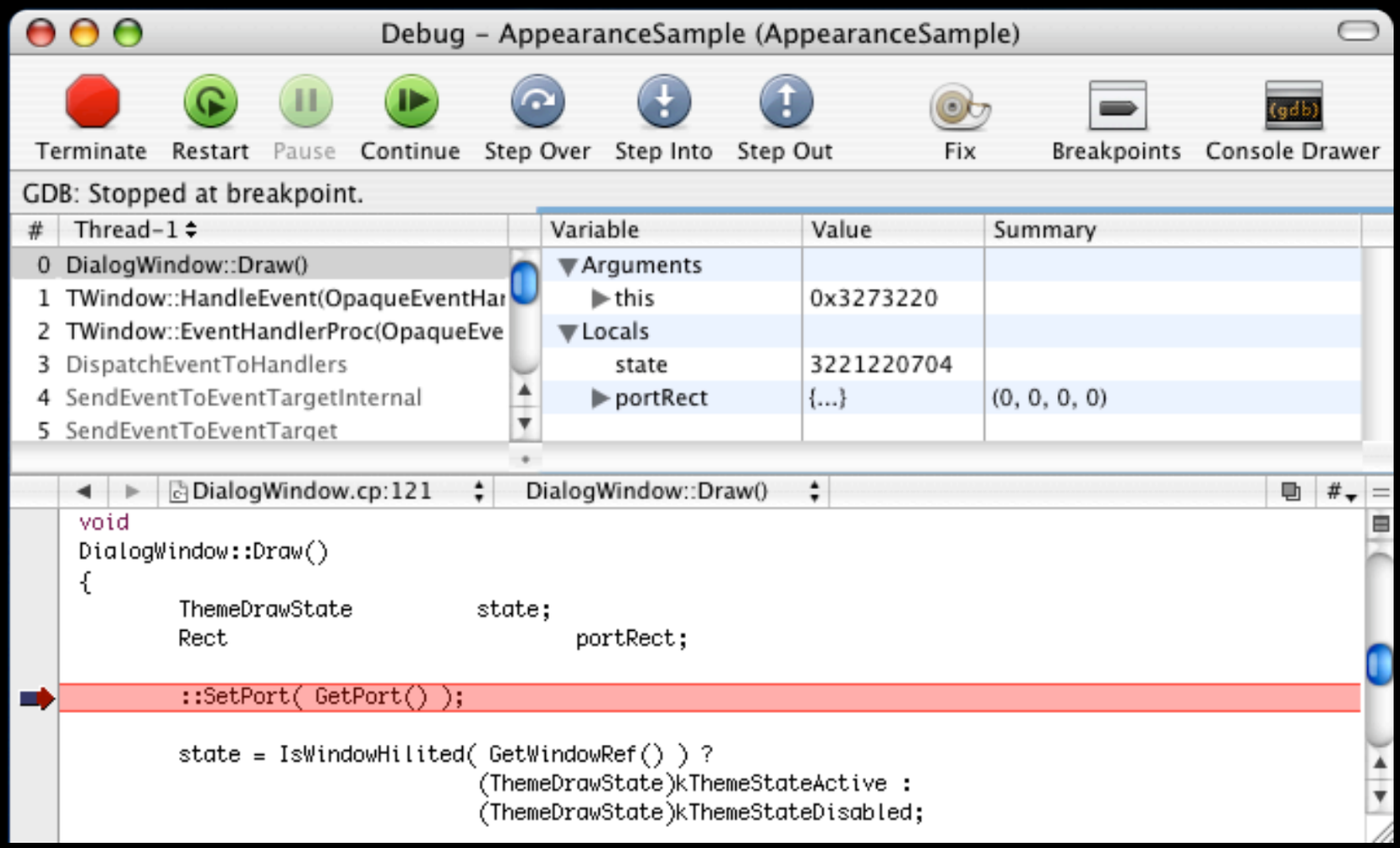


Demo

Debugging With Xcode

Basics of Xcode Debugging

- Debugging in the IDE
- Threads, stacks, and source
- Setting and managing breakpoints



New Debugging Features

- C++ exceptions
- Stepping through templates
- Built-in Carbon and foundation data viewers
- User-defined summary views

Demo

Writing a CustomDataView Formula

- A string with access to a variable and its relatives
- Structs and classes
  - Use %member% for data members
- Expressions
  - Use { } to enclose
  - Evaluates gdb's subset of C/C++ expressions
  - $VAR is the variable being viewed; $PARENT is its owner
- Examples
  - x=%x% y=%y%
  - pstring length={$VAR[0]}

Writing a CustomDataView Bundle

- A bundle with a plist and a shared library
- Expressions in the plist can call library functions
- Library functions execute in the target app's context
  - Don't alter or side-effect anything!
  - Don't leak!
  - Don't tarry!
- Additional value $ID correlates to debugger display line
- Use callbacks for printf and buffer allocation

CustomDataViewer Examples

- plist entry for a handle:
  - `{(char *)DataViewersCarbonHandle($VAR, $ID)}:s`
- Library code for a handle:

```
_pbxgdb_plugin_function_list *xdb = NULL;
#define SSPRINTF (char *)(xdb->message)

char * DataViewersCarbonHandle(Handle handle, int id)
{
    if (handle && IsHandleValid(handle)) {
      return SSPRINTF(id, "size=%ld (%s)", GetHandleSize(handle),
      (HGetState(handle) & kHandleLockedBit ? "lock" : "unlock"));
    }
    return "invalid";
}
```

Demo

Debugging Carbon Events

- Xcode summary view shows event class and kind, but not parameters
- Built-in debugging functions you can call from gdb:
  - `_SetEventTraceEnabled`
  - Available from gdb command line or in Xcode's gdb drawer
- Output goes to StdIO window and console drawer

Debugging Apple Events

- Xcode debugger summary view shows AE DescType and common types, and event class and code
- Set environment variables to trace Apple events
  - AEDebugSends
  - AEDebugReceives
- Built-in debugging functions you can call from gdb:
  - call (void)GDBPrintAEDesc(desc)
  - call (void)GDBPrintHelpDebuggingAppleEvents()

Demo

Tracing Carbon and Apple Events

But I Like MacsBug

- TN2030: GDB for MacsBug veterans
- gdb is worse at:
  - Performance
  - Carbon savvy (dcmds, globals)
  - Seeing the whole machine
- gdb is better at:
  - Symbolics and type-awareness
  - Future and conditional breakpoints
  - Evaluating expressions and calling functions
  - Not crashing the machine during debugging

But I like CodeWarrior IDE Debugging

- CodeWarrior debugging currently uses gdb
- Debugging entry points are available from CW expressions too
- CW memory viewer is still superior and very useful

But I like ZoneRanger

- Everything you know is wrong
- ZoneRanger is a great visualization tool but doesn't address the Mac OS X memory problem
- Use native Mac OS X memory diagnostics to find leaks and diagnose memory problems
  - MallocDebug
  - Memory allocation viewer in Sampler

Memory Diagnostics With Carbon

- NewPtr and NewHandle just call malloc
- Your handles should never move
- Most of your low-memory handling code is useless
- Your big problems will be:
  - Leaks
  - Scribbling on your own allocations
  - Use after release

Wrap Up

Xcode Improves Carbon Debugging

Significant new debugging features

- C++ exceptions
- Stepping through templates
- Built-in Carbon and foundation data viewers
- User-defined summary views

Xcode: A Great Carbon Future

Things we know you're waiting for

- More flexible type viewer
- Generic memory viewer
- Tracing tools built into the IDE
- Faster stepping and viewing
- Watchpoints and eventpoints

Whom to Contact

Godfrey DiGiorgi
Technology Manager, Development Tools
ramarren@apple.com

Development Tools Engineering Feedback
xcode-feedback@group.apple.com

To Post Bug Reports
http://developer.apple.com/bugreporter

Mailing Lists at Apple
http://lists.apple.com

http://developer.apple.com/wwdc2003/urls.html

Roadmap

311 Software Testing Tools for Mac OS X Automate your application testing	**Russian Hill** **Fri., 10:30am**
423 How to Write a Modern Carbon App Getting to know carbon events and Nibs	**Nob Hill** **Fri., 2:00pm**
FF017 Apple Developer Connection Feedback Give us your feedback	**North Beach** **Fri., 2:00pm**

For More Information

Reference Library

Documentation

Debugger Services Reference

ADC Home > Documentation > Carbon > Performance >

Debugging with GDB

ADC Home > Documentation > Tools >

GDB Internals

ADC Home > Documentation > Tools >

Performance Programming Topics

ADC Member Site > Download Software >

Reference Library

Documentation (Cont.)

GDB Release Notes

ADC Home > Documentation > Tools >

Project Builder Help

ADC Home > Documentation > Tools >

Technical Notes

Carbon Tips and Tricks

http://developer.apple.com/carbon/tipsandtricks.html

Reference Library

Other Resources

CHUD Toolkit

ftp://ftp.apple.com/developer /Tool_Chest/Testing_-_Debugging/Performance_tools/

Q&A

Chris Espinosa
Apple Developer Tools
cde@apple.com

http://developer.apple.com/wwdc2003/urls.html

WWDC 2003

Worldwide Developers Conference